NAAN MUDHALVAN PROJECT REPORT

COLLEGE : AALIM MUHAMMED SALEGH COLLEGE OF ENGINEERING

PROJECT TITLE : NEWmart – Grocery Store

TEAM MEMBERS:

1. ARFATH KHAN S (110121104011)
2. IRFAN BASHA J (110121104028)
3. JAMALLUDEEN M (110121104030)
4. HAMEETH S (110121104306)

INTRODUCTION:

NEWmart is your one-stop destination for effortless online shopping. With a user-friendly interface and a comprehensive product catalog, finding the perfectitems has never been easier. Seamlessly navigate through detailed product descriptions, customer reviews, and available discounts to make informed decisions. Enjoy a secure checkout process and receive instant order confirmation. For sellers, our robust dashboard provides efficient order management and insightful analytics to drive business growth. Experience the future of online shopping with EKOmart today.Seamless Checkout Process, Effortless Product Discovery, Personalized Shopping Experience, Efficient Order Management for Sellers, Insightful Analytics for Business Growth.

Pre-requisites for Developing NEWmart Full-Stack Grocery App

To develop a full-stack e-commerce app using **React JS**, **Node.js**, and **MongoDB**, here are the key prerequisites you need to consider:

1. Node.js and npm

Node.js is required to run JavaScript code on the server side. npm (Node Package Manager) is included with Node.js, and it's used to manage project dependencies.

- **Download Node.js:**
  Node.js download page
- **Installation Instructions:**
  Node.js installation guide for various package managers

---

2. MongoDB

You need MongoDB to store the data for products, users, orders, and other e-commerce information. You can install MongoDB locally or use a cloud-based MongoDB service like MongoDB Atlas.

- **Download MongoDB:**
  Download MongoDB Community Edition
- **Installation Instructions:**
  MongoDB installation guide

---

3. Express.js

Express.js is a web application framework for Node.js. It will handle server-side routing, middleware, and API development.

- **Install Express.js:** In your terminal, run the following command to install Express:

  npm install express

---

4. React.js

React.js is a powerful JavaScript library for building interactive user interfaces, and it's perfect for creating dynamic, responsive front-end applications.

- **Follow this guide to install React:**
  React Documentation: Creating a New React App

  To create a new React app, you can use **Create React App**:

  npx create-react-app NEWmart

  cd NEWmart

npm start

---

5. HTML, CSS, and JavaScript

Basic knowledge of HTML, CSS, and JavaScript is essential to structure, style, and add interactivity to the user interface of your e-commerce app.

---

6. Database Connectivity (MongoDB + Node.js)

To connect MongoDB with your Node.js server, you can use **Mongoose**, an Object Data Modeling (ODM) library for MongoDB and Node.js.

- **Install Mongoose:**

  npm install mongoose

For more details on connecting MongoDB to Node.js, check out this guide:
Link: Node.js + Mongoose + MongoDB

---

7. Front-End Framework (React)

Since your project will use React.js for the front-end, you'll be building the user-facing part of the application, including product listings, shopping carts, user profiles, and the admin dashboard using React components.

---

8. Version Control with Git

Use Git for version control to manage your project's codebase, track changes, and collaborate with other developers. GitHub or Bitbucket can be used to host your repository.

- **Download Git:**
  Git download page

---

9. Development Environment (IDE/Code Editor)

Choose a code editor that you're comfortable with. Here are a few popular options:

- **Visual Studio Code:**
  Download Visual Studio Code
- **Sublime Text:**
  Download Sublime Text
- **WebStorm:**
  Download WebStorm

---

10. Set Up the NEWmart Grocery App

To get started with the **NEWmart** Grocery App, follow these steps:

Clone the Repository

1. Open your terminal or command prompt.
2. Navigate to the directory where you want to store the **NEWmart** Grocery App.
3. Clone the repository by executing the following command:

   git clone https://github.com/Arfath02/NN.git

Install Dependencies

1. Navigate into the cloned directory:

   cd NEWmart GroceryApp

2. Install the required dependencies:

   npm install

Start the Development Server

1. To start the development server, run the following command:

npm Start

Or you can run:

npm run dev

2. The app will be accessible by default at:
   http://localhost:5173

   You can change the port by modifying the .env file if needed.

Access the App

1. Open your browser and navigate to http://localhost:5173.
2. You should see the homepage of the **NEWmart** e-commerce app.

---

11. Customization & Development

Once the app is up and running, you can proceed with:

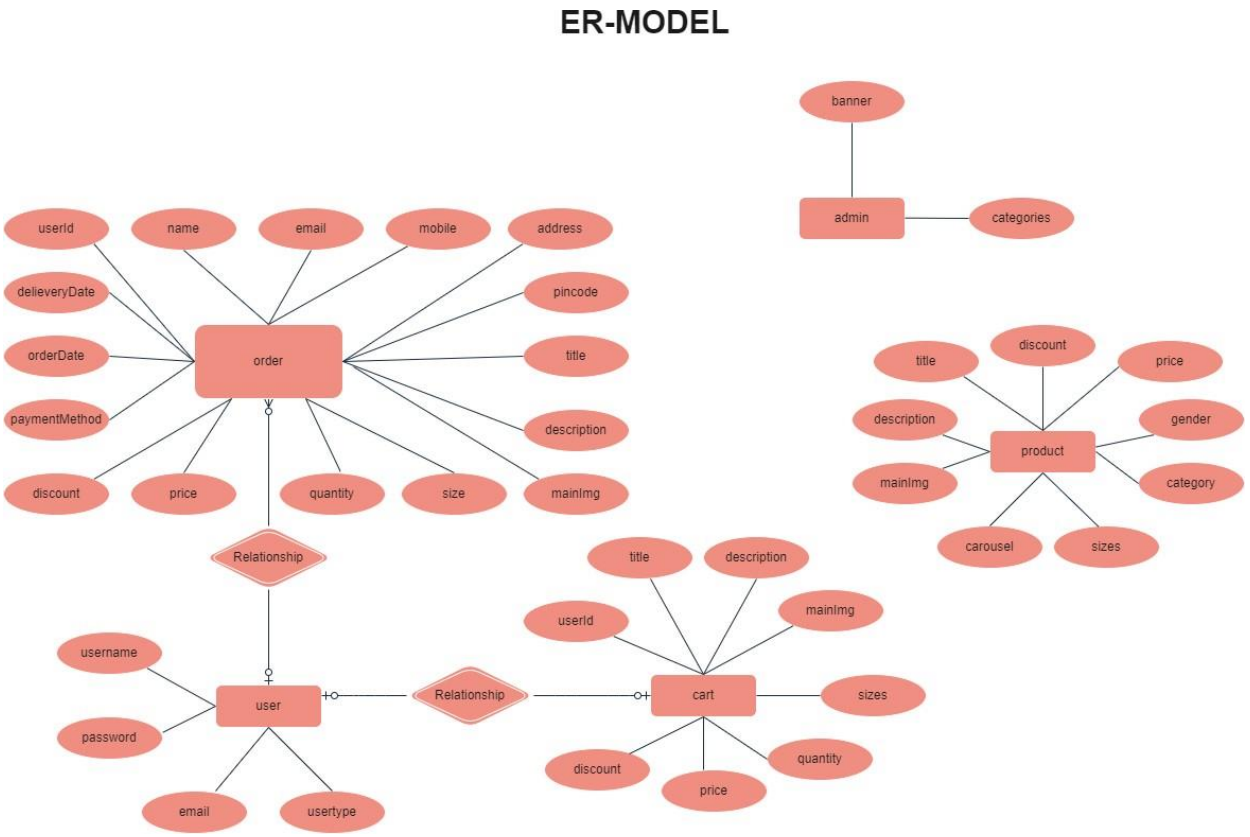
- **Customizing the UI:** Modify the React components to reflect your e-commerce theme (e.g., product listings, shopping cart, checkout process).
- **Building the Backend API:** Use **Express.js** and **MongoDB** to implement features such as product management, order processing, user authentication, and payment integration.
- **Testing:** Test your app locally before deploying it to production.

---

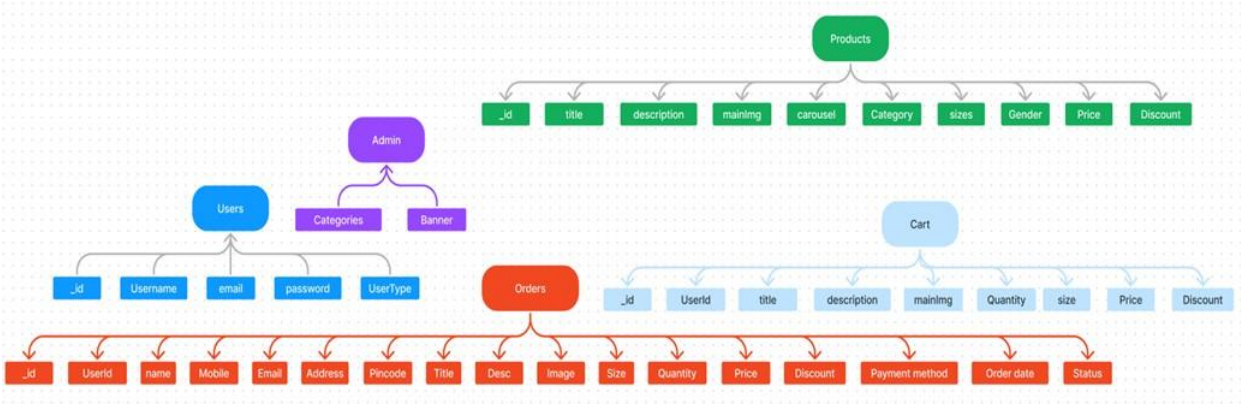
12. Further Customizations

- Customize the front-end to suit your design and branding needs.
- Implement additional features like user authentication (JWT, Passport.js), payment integration (Stripe, PayPal), and admin dashboard for managing orders, products, and users.

ER-DIAGRAM



The Database section represents the database that stores collections for Users, Admin, Cart, Orders and products.



TREND'S E-Commerce Platform ER-Diagram Overview

The **TREND'S** ER-diagram (Entity-Relationship Diagram) represents the various entities within the platform and how they are interconnected. It illustrates the relationships between **Users**, **Products**, **Cart**, **Orders**, and **Admin** entities. Here's a breakdown of these entities and their relationships:

Entities and Their Relationships

1. **USER**
    - **Description**: Represents the individuals or entities who are registered on the **TREND'S** platform.
    - **Key Fields**: User ID, Name, Email, Password, Address, Phone Number, etc.
    - **Relationships**:
        - Users can **add products to the Cart**.
        - Users can **place Orders**.
        - Users can **write Reviews** for products.
        - Users have a **history of Orders** and **saved addresses**.
2. **ADMIN**
    - **Description**: Represents the administrative collection that contains important details such as banner images, product categories, and site-wide settings.
    - **Key Fields**: Admin ID, Banner Image, Categories, etc.
    - **Responsibilities**:
        - Manage product **categories**.
        - Update the **home page banners**.
        - Oversee product listings and **approve new sellers**.

3. **PRODUCTS**
    - **Description**: Represents a collection of all the products available on the **TREND'S** platform.
    - **Key Fields**: Product ID, Name, Price, Description, Stock Quantity, Images, Seller ID, Category, etc.
    - **Relationships**:
        - Products can be added to the **Cart** by users.
        - Products are part of **Orders** placed by users.
        - Products can have **Reviews** from users.
4. **CART**
    - **Description**: This collection stores the products that have been added to the cart by users before completing a purchase. Each cart is uniquely identified by the **user ID**.
    - **Key Fields**: Cart ID, User ID, Product ID(s), Quantity, etc.
    - **Relationships**:
        - Cart items are associated with a **specific user**.
        - When a user proceeds with the checkout, the cart items become part of an **Order**.
5. **ORDERS**
    - **Description**: This collection stores all the orders made by users on the platform.
    - **Key Fields**: Order ID, User ID, Product IDs, Quantity, Total Price, Shipping Address, Payment Method, Status (Pending/Completed), etc.
    - **Relationships**:
        - Each order is placed by a **User**.

- Orders consist of **Products** from the cart that have been purchased.
- Orders are associated with a **shipping address** and **payment method**.

Key Features of TREND'S E-Commerce Platform

1. **Comprehensive Product Catalog**
   TREND'S boasts an extensive catalog of products, offering a diverse range of items from various categories. Shoppers can easily browse through detailed product listings, complete with descriptions, customer reviews, prices, and available discounts. This helps users find the best items for their needs and make informed decisions.
2. **Shop Now Button**
   Each product listing features a "Shop Now" button. When you find a product that fits your preferences, simply click on the button to begin the purchasing process.
3. **Order Details Page**
   Upon clicking the "Shop Now" button, users are redirected to an **Order Details Page**. Here, users can provide relevant information such as:
   - **Shipping Address**
   - **Preferred Payment Method**
   - Any specific **product requirements** (such as size, color, etc.)
4. **Secure and Efficient Checkout Process**
   TREND'S ensures that your personal information is securely handled. The checkout process is streamlined to make the purchasing experience as fast and seamless as possible. Payments and sensitive information are processed using industry-standard encryption protocols to ensure security.
5. **Order Confirmation and Details**
   After placing an order, users receive a confirmation notification. The **Order**

Details Page will display all the necessary information about the order, including:

- **Shipping Details**
- **Payment Method**
- **Product Specifications**
- Any **custom requests** made during the order process.

Seller Dashboard

In addition to providing an excellent experience for shoppers, **TREND'S** includes a robust **Seller Dashboard**, offering sellers the tools they need to efficiently manage their products and sales. Some key features include:

- **Product Management**: Sellers can add, update, and manage multiple product listings.
- **Order History**: Sellers can view their entire order history, track shipments, and monitor customer activity.
- **Sales Analytics**: Sellers can access detailed reports and analytics to help them monitor sales, revenue, and customer engagement.
- **Customer Interaction**: Sellers can manage customer messages, answer queries, and monitor feedback or reviews related to their products.

User Experience

TREND'S is designed to enhance the online shopping experience by providing:

- **A wide variety of products**: From fashion to electronics, users can find everything they need in one place.
- **Easy navigation**: The user interface is intuitive, allowing customers to browse, search, and filter products with ease.

- **Fast and secure checkout**: Whether using a credit card, PayPal, or another payment method, **TREND'S** makes the checkout process smooth and reliable.
- **Comprehensive seller tools**: Sellers have access to a full suite of tools to help manage and grow their business on the platform.

By offering a seamless and user-friendly shopping experience, **TREND'S** ensures both shoppers and sellers enjoy a convenient and efficient platform for online shopping.

APPLICATION FLOW

USER & ADMIN FLOW:

1. User Flow:

• Users start by registering for an account.

• After registration, they can log in with their credentials.

• Once logged in, they can check for the available products in the platform. • Users can add the products they wish to their carts and order.

• They can then proceed by entering address and payment details. • After ordering, they can check them in the profile section.

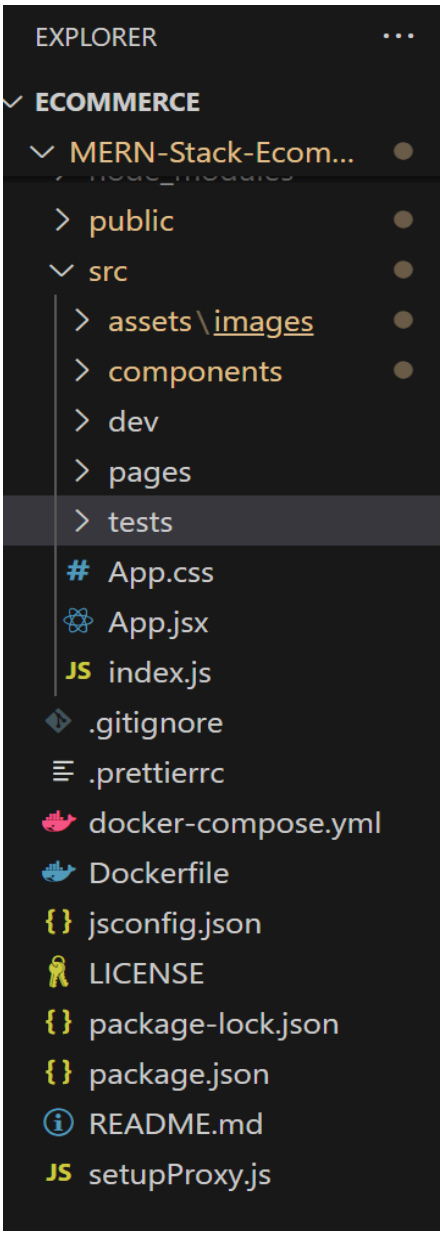
2. Admin Flow:

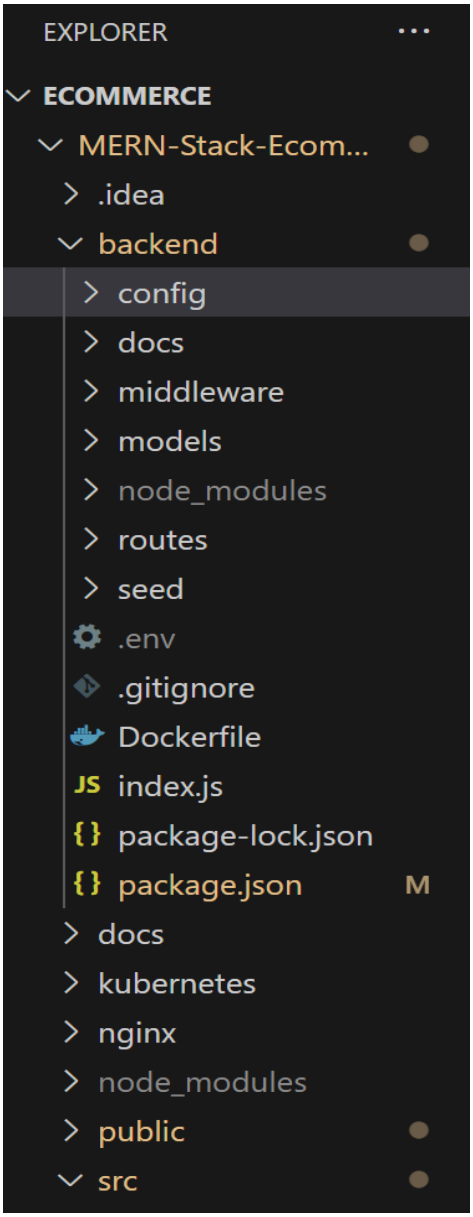
• Admins start by logging in with their credentials.

• Once logged in, they are directed to the Admin Dashboard.

• Admins can access the users list, products, orders, etc.,

PROJECT STRUCTURE:





This structure assumes a React app and follows a modular approach. Here's a brief explanation of the main directories and files:

• src/components: Contains components related to the application such as, register, login, home, etc.,

• src/pages has the files for all the pages in the application.

Project Setup And Configuration:

Milestone 1: Project Setup and Configuration:

1. Install required tools and software:

• Node.js.

Reference Article: https://www.geeksforgeeks.org/installation-of-node-js-on-windows/

• Git.

Reference Article: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

2. Create project folders and files:

• Client folders.

• Server folders

Backend Development

1. Setup express server:

• Create index.js file.

• Create an express server on your desired port number.

• Define API's

Now your express is successfully created.

Set Up Project Structure:

• Create a new directory for your project and set up a package.json file using the npm init command.

• Install necessary dependencies such as Express.js, Mongoose, and other required packages.

2. Database Configuration:

• Set up a MongoDB database either locally or using a cloud-based MongoDB service like MongoDB Atlas or use locally with MongoDB compass.

• Create a database and define the necessary collections for admin, users, products, orders and other relevant data.

3. Create Express.js Server:

• Set up an Express.js server to handle HTTP requests and serve API endpoints.

• Configure middleware such as body-parser for parsing request bodies and cors for handling cross-origin requests.

4. Define API Routes:

• Create separate route files for different API functionalities such as users, orders, and authentication.

• Define the necessary routes for listing products, handling user registration and login,managing orders, etc.

• Implement route handlers using Express.js to handle requests and interact with the database.

5. Implement Data Models:

• Define Mongoose schemas for the different data entities like products, users, and orders.

• Create corresponding Mongoose models to interact with the MongoDB database.

• Implement CRUD operations (Create, Read, Update, Delete) for each model to perform database operations.

6. User Authentication:

• Create routes and middleware for user registration, login, and logout.

• Set up authentication middleware to protect routes that require user authentication.

7. Handle new products and Orders:

• Create routes and controllers to handle new product listings, including fetching products data from the database and sending it as a response.

• Implement ordering(buy) functionality by creating routes and controllers to handle order requests, including validation and database updates.

8. Admin Functionality:

• Implement routes and controllers specific to admin functionalities such as adding products, managing user orders, etc.

• Add necessary authentication and authorization checks to ensure only authorized admins can access these routes.

9. Error Handling:

• Implement error handling middleware to catch and handle any errors that occur during the API requests.

• Return appropriate error responses with

Database Development:

Create database in cloud

• Install Mongoose.

• Create database connection.

Reference Article: https://www.mongodb.com/docs/atlas/tutorial/connect-to-your-cluster/

Schema use-case:

1. User Schema:

• Schema: userSchema

• Model: 'User'

• The User schema represents the user data and includes fields such as username, email, and password.

• It is used to store user information for registration and authentication purposes.

• The email field is marked as unique to ensure that each user has a unique email address

2. Product Schema:

• Schema: productSchema

• Model: 'Product'

• The Product schema represents the data of all the products in the platform.

• It is used to store information about the product details, which will later be useful for ordering .

3. Orders Schema:

• Schema: ordersSchema

• Model: ‘Orders’

• The Orders schema represents the orders data and includes fields such as userId, product Id, product name, quantity, size, order date, etc.,

• It is used to store information about the orders made by users.

• The user Id field is a reference to the user who made the order.

4. Cart Schema:

• Schema: cartSchema

• Model: ‘Cart’

• The Cart schema represents the cart data and includes fields such as userId, product Id, product name, quantity, size, order date, etc.,

• It is used to store information about the products added to the cart by users. • The user Id field is a reference to the user who has the product in cart.

5. Admin Schema:

• Schema: adminSchema

• Model: ‘Admin’

• The admin schema has essential data such as categories, banner.

Frontend development

1. Setup React Application:

- Create a React app in the client folder.
- Install required libraries
- Create required pages and components and add routes.

2.Design UI components:

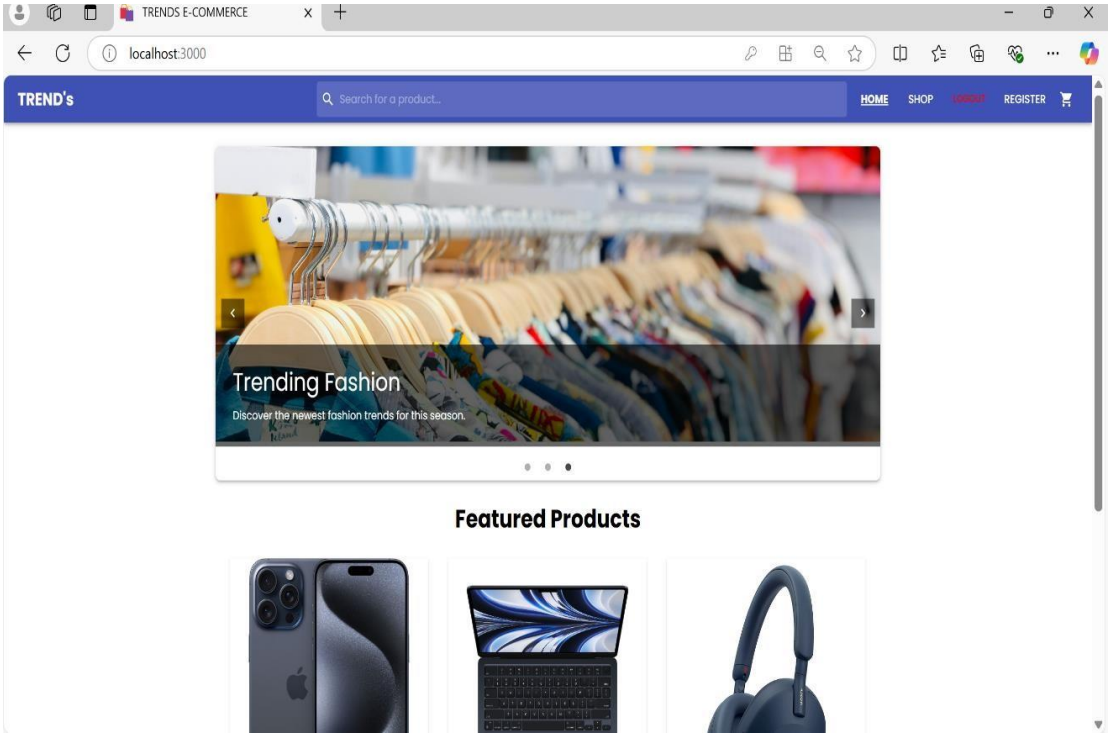
- Create Components.
- Implement layout and styling.
- Add navigation.

3.Implement frontend logic:

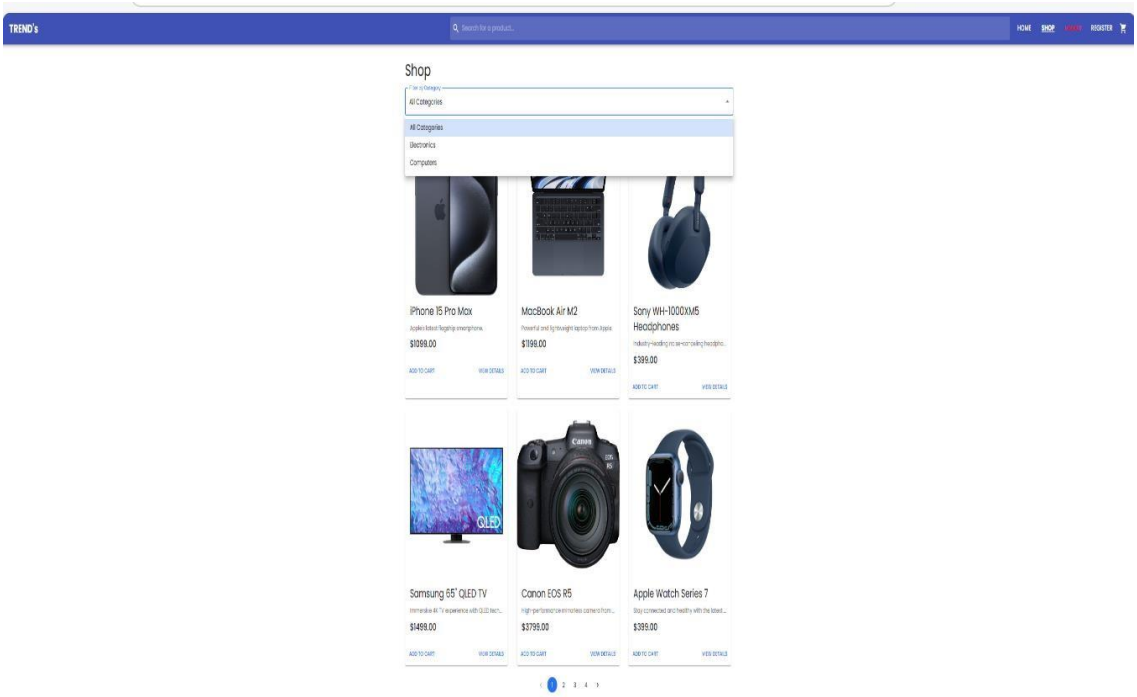
- Integration with API endpoints.
- Implement data binding.

OUTPUT:

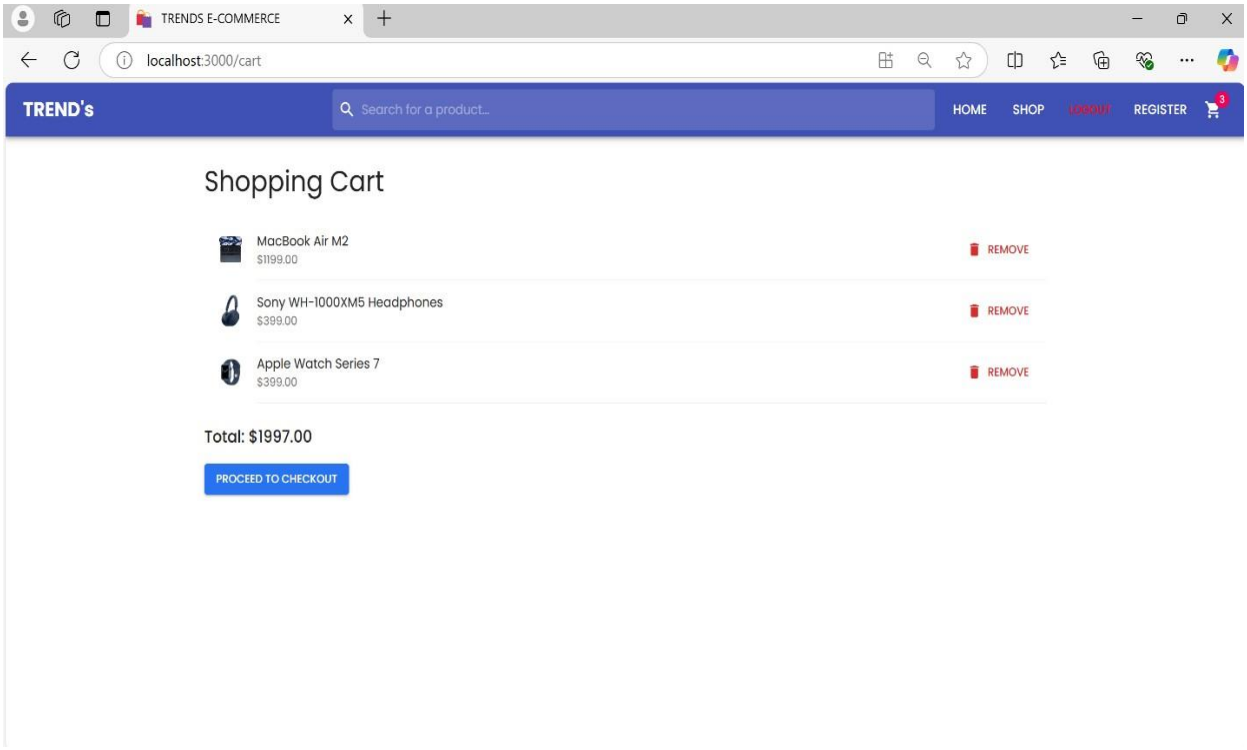
HOME PAGE



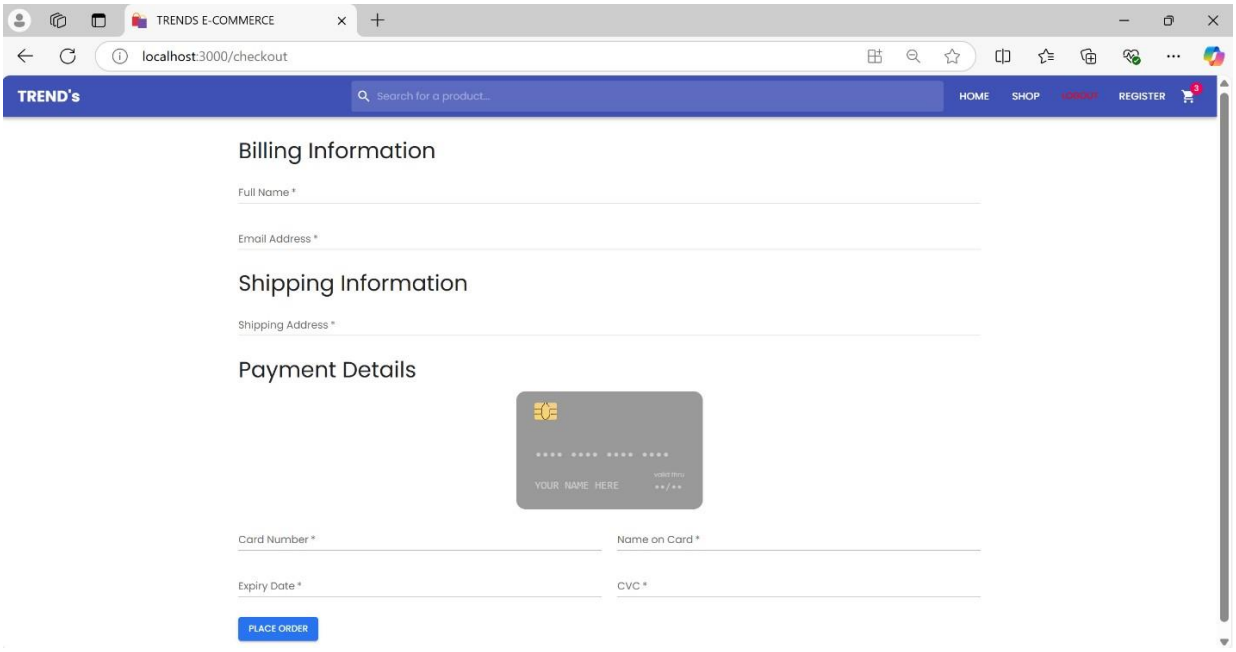
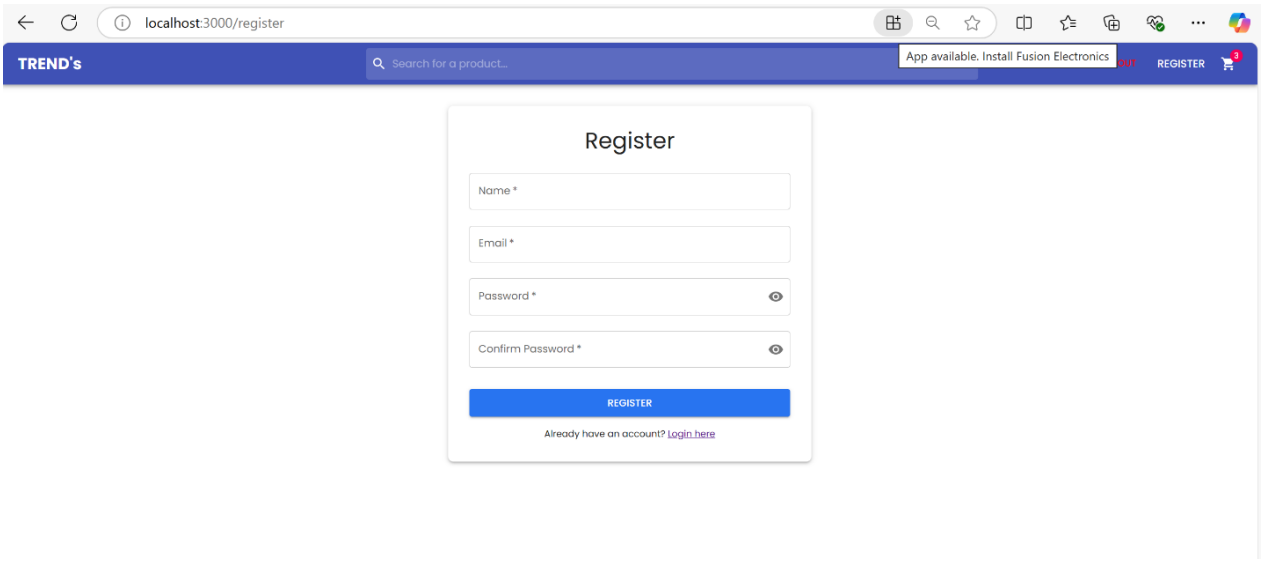
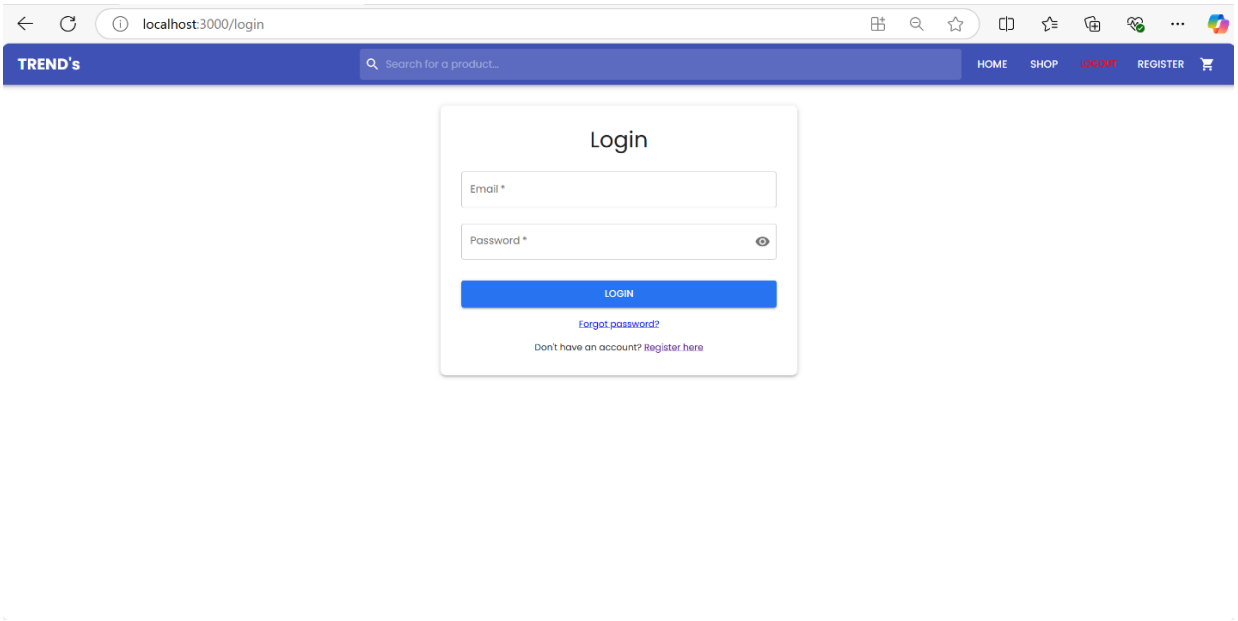
PRODUCTS


TREND's
HOME
SHOP
REGISTER
Shop
All Categories
All Categories
Electronics
Computers
iPhone 15 Pro Max
$1099.00
MacBook Air M2
$1199.00
Sony WH-1000XM5 Headphones
$399.00
Samsung 65" QLED TV
$1499.00
Canon EOS R5
$3799.00
Apple Watch Series 7
$399.00

SHOPPING CART



PAYMENT PAGE



REGISTER / LOGIN PAGE